



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/051,268	01/18/2002	Sundeep Chandhoke	5150-58300	8985
35690	7590	03/03/2008		
MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL, P.C.			EXAMINER	
P.O. BOX 398			PHAM, CHRYSTINE	
AUSTIN, TX 78767-0398			ART UNIT	PAPER NUMBER
			2192	
			MAIL DATE	DELIVERY MODE
			03/03/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.



UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

MAILED

MAR 03 2003

Technology Center 2100

**BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES**

Application Number: 10/051,268
Filing Date: January 18, 2002
Appellant(s): CHANDHOKE ET AL.

Jeffrey C. Hood (Reg. No. 35198)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 09/01/2006 appealing from the Office action mailed 11/28/2005.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,226,783	LIMONDIN ET AL.	5-2001
5,966,532	MCDONALD ET AL.	10-1999

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

- ❖ Claims 1, 4-11, 18-28, 31-45 are rejected under 35 U.S.C. 102(e) as being anticipated by Limondin et al. (US 6226783), hereinafter, *Limondin et al.*.

Claim 1

Limondin et al. teach a computer-implemented method (e.g., see *object oriented method* col.2:59-65) for creating a graphical (i.e., graphical data flow) program (e.g., see *machine vision application* col.1:15-35; see *structuring a software step program, vision applications, computer programs, user interfaces* col.1:55-67; see *programs, graphical user interface* col.2:1-3; see *user interface components, GUI* col.2:15-22; col.2:59-65; see *STEPS, 100 FIG.2 & associated text; see FIG.6 & associated text; see FIGS.8A-8C & associated text; see step program, editor objects* col.7:35-44) based on a sequence (i.e., prototype) (e.g.,

see *series of operations*, “*steps*” Abstract; see *set of operations*, *steps*, *sequence* col.1:15-35; see *list of operations* col.3:1-10; see *step inputs* col.5:8-13; see *execution order* col.5:20-35) that includes motion control (e.g., see *WarpStep* 122 FIG.2 & associated text; see *WARP* 122 FIG.3 & associated text), machine vision (e.g., see *VISION PROCESSOR BOARD* FIG.8A & associated text; see *machine vision application* col.1:15-35; see *machine vision operations* col.1:55-67), and data (i.e., images, and measurement data) acquisition (DAQ) operations (i.e., functionality) (e.g., see *AcquireStep* 112 FIG.2 & associated text; see *ACQUIRE* 112 FIG.3 & associated text; see *acquisition of an input image* col.1:15-35; see *feature extraction*, *features*, *size*, *area*, *length*, *distance* col.1:15-35;), the method comprising:

- o means for displaying a graphical user interface (GUI) that provides GUI access to a set of operations (e.g., see *user interfaces* col.1:55-67; see *step program*, *tree control window* col.4:50-55; see *iconic picture*, *individual step operation* col.5:35-45; see FIG.2 & associated text; see *editor objects*, *graphical view*, *step* col.6:59-65), wherein the set of operations includes one or more motion control operations (e.g., see *motion device* col.3:15-27), one or more machine vision (i.e., image analyzing) operations (e.g., see *accelerated image processing* col.2:40-50), and one or more DAQ operations (e.g., see *AcquireStep* 706a, *AcquireStep* 706b FIG.9 & associated text; see *image acquisition* col.2:40-50);

- means for receiving user input to the graphical user interface specifying the sequence of operations (e.g., see *series of operations*, “*steps*” Abstract; see *set of operations*, *steps*, *sequence* col.1:15-35; see *list of operations* col.3:1-10; see *step inputs* col.5:8-13; see *operator*, *user interface control*, *step parameters*, *settings* col.6:59-col.7:3), wherein the specified sequence of operations includes at least one motion control operation (e.g., see *WarpStep 122* FIG.2 & associated text; see *WARP 122* FIG.3 & associated text), at least one machine vision operation (e.g., see *VISION PROCESSOR BOARD* FIG.8A & associated text; see *machine vision application* col.1:15-35; see *machine vision operations* col.1:55-67), and at least one DAQ operation (e.g., see *AcquireStep 112* FIG.2 & associated text; see *ACQUIRE 112* FIG.3 & associated text; see *acquisition of an input image* col.1:15-35; see *feature extraction*, *features*, *size*, *area*, *length*, *distance* col.1:15-35;);
- means for storing the specified sequence of operations based on the user input (e.g., see *GUID*, *database* col.4:55-65; see *steps*, *data structure*, *map* col.6:34-40; see *step program*, *disk* col.10:5-17); and
- means for programmatically/automatically generating a graphical program to implement the specified sequence of operations, wherein said automatically generating the graphical program comprises generating graphical code in the graphical program without direct user input (e.g., see *execution order*, *step*, *attribute* col.5:25-35; see *step program*, *steps*,

software objects, C++ classes col.4:25-67; see iconic picture, step operation, step program 100 col.5:37-40; col.2:59-67; col.4:34-67), and wherein the graphical code comprises a plurality of interconnected nodes (i.e., linking portions of graphical code wherein each portion of graphical code implements one of the operations in the sequence) which visually indicate the functionality of the graphical program that visually indicate functionality of the graphical program (e.g., see software objects, steps, connection, inputs, outputs col.2:59-65; see STEPS, 100 FIG.2 & associated text; see STEPS, 706a, 706b, 714 FIG.9 & associated text; see steps, datum objects col.4:34-42; see iconic picture, individual step operation col.5:35-45; see FIG.2 & associated text).

Claim 4

The rejection of base claim 1 is incorporated. *Limondin et al.* further teach further comprising: executing the graphical program to perform (i.e., affect an action which operation is operable to perform) the sequence of operations (e.g., see *target processor, user program Abstract; col.2:30-35; see execution, program col.2:40-50; see steps, operations col.2:59-67; see execution order col.5:20-35; col.10:4-10).*

Claim 5

The rejection of base claim 1 is incorporated. *Limondin et al.* further teach wherein the graphical program includes a block diagram portion (e.g., see *software*

objects, steps, tree structure col.4:25-31; see *STEPS 100* FIG.2 & associated text) and a user interface panel portion (e.g., see *step program, tree control window* col.4:50-55; see FIG.2 & associated text).

Claim 6

The rejection of base claim 1 is incorporated. Claim recites limitations, which have been addressed in claim 1, therefore, is rejected for the same reasons as cited in claim 1.

Claim 7

The rejection of base claim 1 is incorporated. *Limondin et al.* further teach wherein said programmatically generating the graphical program comprises including one or more nodes in the graphical program corresponding to the operations in the sequence (e.g., see *iconic picture, individual step operation* col.5:35-45; see FIG.2 & associated text; see *step code, object functions* col.10:9-17).

Claim 8

The rejection of base claim 1 is incorporated. Claim recites limitations, which have been addressed in claim 3, therefore, is rejected for the same reasons as cited in claim 3.

Claim 9

The rejection of base claim 8 is incorporated. *Limondin et al.* further teach wherein each portion of graphical code includes one or more graphical program nodes, wherein each node has one or more inputs or outputs (e.g., see *IN DATA 510, 512, 514, OUT DATA 234, 236, 238 FIG.5 & associated text*); wherein generating each portion of graphical code comprises connecting the node inputs and outputs together in order to implement the operation with which the portion of graphical code is associated (e.g., see *STEP OBJECT 200 FIG.5 & associated text*; see *connection of inputs to outputs at each step col.2:59-65*; see *inputs, outputs, operation, step col.3:1-11*).

Claim 10

The rejection of base claim 8 is incorporated. *Limondin et al.* further teach wherein linking a first portion of graphical code to a second portion of graphical code comprises connecting an output of a node in the first portion of graphical code to an input of a node in the second portion of graphical code (e.g., see *connection of inputs to outputs, between steps col.2:59-65*; see *results, output, inputs, one step, other steps col.4:42-50*).

Claim 11

The rejection of base claim 8 is incorporated. *Limondin et al.* further teaches further comprising: for each operation in the sequence, retrieving information

associated with the operation from a database; wherein generating the portion of graphical code that implements a particular operation utilizes the database information retrieved for the particular operation (e.g., see *recipe database* col.2:1-6; see *GUID, database* col.4:55-65).

Claim 18

The rejection of base claim 1 is incorporated. *Limondin et al.* further teach wherein said receiving user input to the graphical user interface specifying a desired sequence of operations does not include receiving user input specifying programming language code to implement the sequence of operations (e.g., see *user programs, computer language, programming language* Abstract; see *drop-in functionality, language independent software components* col.1:55-67; see *program, point and click graphical user interface* col.2:1-6).

Claim 19

The rejection of base claim 1 is incorporated. *Limondin et al.* further teach wherein the sequence is operable to perform one or more (i.e., two) of:

- control motion of a device (e.g., see *motion device, moving camera* col.3:15-27);
- analyze acquired images (e.g., see *acquisition of an input image* col.1:15-35; see *accelerated image processing, image acquisition* col.2:40-50; see *camera images* col.4:15-22; see FIG.9 & associated text; see FIG.3 & associated text; see *acquisition of an image* col.5:60-65; see *camera images* col.7:64-67); and

- acquire measurement data (e.g., see *acquisition of an input image* col.1:15-35; see *feature extraction, features, size, area, length, distance* col.1:15-35; see *distance, features* col.4:15-22; see FIG.9 & associated text; see *camera images, distance between features* col.7:64-67; col.9:5-25).

Claims 20-23

The rejection of base claim 1 is incorporated. Claims recite limitations, which have been addressed in claims 1, 4, 18, 19, therefore, are rejected for the same reasons as cited in claims 1, 4, 18, 19.

Claim 24

The rejection of base claim 22 is incorporated. *Limondin et al.* further teach further comprising: for each operation to be configured, displaying a graphical panel including graphical user interface elements for setting properties of the operation and receiving user input to the graphical panel to set one or more properties of the operation (e.g., see *user interface components, GUI, setting of parameters* col.2:15-22; see *parameters, inputs, outputs* col.3:1-15; col.4:55-65).

Claims 25-27

Claims recite limitations, which have been addressed in claim 1, therefore, are rejected for the same reasons as cited in claim 1.

Claim 28

Claim recites a memory medium comprising program instructions executable (e.g., see *step program*, *disk* col.10:5-17) for performing the method addressed in claim 1, therefore, is rejected for the same reasons as cited in claim 1.

Claims 31-36

Claims recite limitations, which have been addressed in claims 1-4, 19, and 20, therefore, are rejected for the same reasons as cited in claims 1-4, 19, and 20.

Claim 37

Limondin et al. teach a system (e.g., see FIGS.8A-8C & associated text) for creating a graphical program based on a sequence that includes motion control, machine vision, and data acquisition (DAQ) operations (see claim 1), the system comprising:

- a processor (e.g., see *target processor* Abstract);
- a memory storing program instructions (e.g., see *step program*, *disk* col.10:5-17);
- and
- a display device (e.g., see FIG.2 & associated text);

Art Unit: 2192

- wherein the processor is operable to execute the program instructions stored in the memory to:
 - display a graphical user interface (GUI) on the display device that provides access to a set of operations, wherein the set of operations includes one or more motion control operations, one or more machine vision operations, and one or more DAQ operations (see claim 1);
 - receive user input to the graphical user interface specifying the sequence of operations, wherein the specified sequence of operations includes at least one motion control operation, at least one machine vision operation, and at least one DAQ operation (see claim 1);
 - store the specified sequence of operations based on the user input (see claim 1); and
 - programmatically/automatically generate a graphical program to implement the specified sequence of operations, wherein, in automatically generating the graphical program, the program instructions are executable to generate graphical code in the graphical program without direct user input, wherein the graphical code comprises a plurality of interconnected nodes which visually indicate the functionality of the graphical program (see claim 1).

Claim 38

The rejection of base claim 37 is incorporated. *Limondin et al.* further teach further comprising:

- a motion control device (e.g., see *motion device, moving camera* col.3:15-27; see *Calibrate* col.7:19-50);
- an image acquisition device (e.g., see *acquisition of an input image* col.1:15-35);
and
- a data acquisition device (e.g., see *acquisition of an input image* col.1:15-35);
- wherein the processor is operable to execute the graphical program to:
 - control the motion control device to move an object (e.g., see *WarpStep* 122 FIG.3 & associated text; see *WarpPart, move the image* col.7:19-50);
 - control the image acquisition device to acquire one or more images of the object (e.g., see *acquisition of an input image* col.1:15-35; see *accelerated image processing, image acquisition* col.2:40-50; see *camera images* col.4:15-22; see FIG.9 & associated text; see FIG.3 & associated text; see *acquisition of an image* col.5:60-65; see *camera images* col.7:64-67); and
 - control the data acquisition device to acquire measurement data of the object (e.g., see *acquisition of an input image* col.1:15-35; see *feature extraction, features, size, area, length, distance* col.1:15-35; see *distance, features* col.4:15-22; see FIG.9 & associated text; see *camera images, distance between features* col.7:64-67; col.9:5-25).

Claim 39

Claim recites limitations, which have been addressed in claims 1, and 37, therefore, is rejected for the same reasons as cited in claims 1, and 37.

Claims 40-45

Claims recite limitations, which are subcombinations that have been addressed in claim 1, therefore, are rejected for the same reasons as cited in claim 1.

- ❖ Claims 12-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over unpatentable over *Limondin et al.* in view of McDonald et al. (US 5966532), hereinafter, *McDonald et al.*.

Claim 12

The rejection of base claim 1 is incorporated. *Limondin et al.* do not expressly disclose creating an association between the sequence and the graphical program, modifying the sequence to create a new sequence in response to user input after said creating the association, and modifying the graphical program according to the new sequence to create a new graphical program. However, *McDonald et al.* disclose:

- creating an association between the sequence and the graphical program (e.g., see 212 FIG.2 & associated text; see 212 FIG.4 & associated text; see

Art Unit: 2192

association, control, graphical code portion col.5:10-25; col.3:60-col.4:20; col.4:1-10),

- modifying the sequence to create a new sequence in response to user input after said creating the association (e.g., see 226-230 FIG.3 & associated text; see 228 FIG.4 & associated text; col.5:20-45), and
- modifying the graphical program according to the new sequence to create a new graphical program (e.g., see 226-230 FIG.3 & associated text; see 230 FIG.4 & associated text).

Limondin et al. and *McDonald et al.* are analogous art because they are both directed to generating graphical program based on user inputs specifying data acquisition operations. It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.* for the inclusion of creating an association between the sequence and the graphical program, modifying the sequence, and modifying the graphical program to create new graphical program. And the motivation for doing so would have been to facilitate changes or updates to graphical code portions and accordingly, the associations between user inputs (i.e., graphical icons representing corresponding operations) and the aforementioned graphical code portions, thus, providing an improved graphical programming environment where the user has ultimate programming control over the how the graphical programs are generated or defined based input sequences received through a GUI (see *McDonald et al.* col.3:35-57; col.11:55-63; col.13:42-58; col.14:25-42).

Claim 13

The rejection of base claim 12 is incorporated. *Limondin et al.* *McDonald et al.* further teach

- wherein said modifying the graphical program according to the new sequence uses the association between the sequence and the graphical program (e.g., see 224 FIG.3 & associated text; see 282, 284 FIG.6 & associated text; col.6:1-10);
- wherein the association remains between the new sequence and the new graphical program (e.g., see 288 FIG.6 & associated text). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.*. And the motivation for doing so would have been that which has been recited in claim 12.

Claim 14

The rejection of base claim 1 is incorporated. *McDonald et al.* further teach further comprising:

- creating an association between the sequence and the graphical program (see claim 12); and

Art Unit: 2192

- locking the association between the sequence and the graphical program, wherein said locking prevents user editing of the graphical program (e.g., see *locking prevents user editing* col.5:10-22). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.*. And the motivation for doing so would have been that which has been recited in claim 12.

Claim 15

The rejection of base claim 14 is incorporated. *McDonald et al.* further teach further comprising:

- unlocking the association between the sequence and the graphical program in response to user input after said locking (e.g., see 262 FIG.5 & associated text);
- directly changing the graphical program in response to user input after said unlocking (e.g., see 266 FIG.5 & associated text). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.*. And the motivation for doing so would have been that which has been recited in claim 12.

Claim 16

The rejection of base claim 15 is incorporated. *McDonald et al.* further teach wherein said unlocking removes the association between the sequence and the graphical program (e.g., see 264 FIG.5 & associated text). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.*. And the motivation for doing so would have been that which has been recited in claim 12. It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.*. And the motivation for doing so would have been that which has been recited in claim 12.

Claim 17

The rejection of base claim 14 is incorporated. *McDonald et al.* further teach further comprising:

- modifying the graphical program in response to user input after said generating the graphical program and after said creating the association between the sequence and the graphical program (e.g., col.5:10-62);
 - determining if an association exists between the sequence and the graphical program in response to said modifying the graphical program (e.g., see 224 FIG.3 & associated text; see *wizard* col.5:35-40); and
- removing the association between the sequence and the graphical program in response to said modifying (e.g., col.5:47-63; see *breaks the*

association col.6:5-10). It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of *McDonald et al.* into that of *Limondin et al.*. And the motivation for doing so would have been that which has been recited in claim 12.

(10) Response to Argument

Argument 1 (with respect to Claims 1, 25, 26, 28, 37, 39, 41-45): 'Limondin does not teach automatically generating the step program or generating "graphical code" in the step program *without direct user input*' (Brief, page 17, last paragraph)(Emphasis added).

Response to Argument 1: As has been clearly established in the Final Office Action mailed on 11/28/2005 (see page 4), it should be noted that claim 1 recites "...*automatically .. generating graphical code ... without direct user input*" **after** reciting "*receiving user input ... specifying sequence of operations..*". It is clear from the plain language of claim 1 that the sole purpose of the graphical program is to implement a *sequence of operations*. It is clear from the plain language of claim 1 that said *sequence of operations is specified by the user via user input* to the graphical user interface. In other words, the user input specifying the sequence of operations is an indirect user input, without which, the graphical program cannot be generated. Thus, the claimed limitation "*without direct user input*" in the "*automatically generating a graphical program to implement the sequence of operations ... without direct user input*"

clause merely amounts and/or is the equivalence of "***with indirect user input***" because the generation of graphical program depends on indirect user input, specifying the sequence of operations to be implemented by the graphical program. As has been established in the Final Office Action mailed on (11/28/2005), the *step program* is composed of steps (i.e., software objects or C++ classes) (see at least *step program, steps, software objects, C++ classes* col.4:25-67, Limondin). Limondin specifically states that the steps can be *edited graphically* using a point-and-click *user interface* (i.e., GUI) and that the step program is displayed graphically in a tree control window (see at least col.4:25-67). FIG.2 of Limondin is user interface view of the step program. Limondin further discloses the individual step operation/functionality of the step program is represented [graphically] by an iconic picture (see at least *iconic picture, step operation, step program 100* col.5:37-40). Limondin teaches the step program is created and/or edited by having the user graphically manipulating/editing the step icons' parameters (i.e., inputs and outputs) (see at least *results, outputs, inputs, steps* col.4:42-50) and connections between the icons and execution order (i.e., sequence and/or control flow) of the steps (i.e., operations) can also be graphically defined by the user (see at least *execution order, step, attribute* col.5:25-35). It is respectfully submitted that the user's graphical manipulation of step icons' inputs and outputs and their execution order anticipates "user input to the GUI specifying the sequence of operations". Although Limondin's graphical step program is modified in response to the user input, it should be understood that the actual programming/software code for the step program (i.e., graphical code for the step program) is automatically generated to

Art Unit: 2192

reflect (i.e., implement) the user's modification (i.e., specified sequence of operations). In other words, there is no direct user input (i.e., actual programming/software code) during the automatic generation of the graphical step program.

Argument 2 (with respect to Claims 1, 25, 26, 28, 37, 39, 41-45): 'Limondin does not teach at least the limitation of "automatically generating a graphical program based on the prototype"' (Brief, page 19, 2nd to last paragraph).

Response to Argument 2: It should be noted that the Specification uses the term "prototype" and the verb "develop" interchangeably, suggesting that "prototype" is defined as "to develop" (Specification, page 2, lines 19-20). On page 9, lines 8-9, the Specification also defines "prototype" to be "a sequence of motion control operations". Needless to say, Limondin's user's graphical manipulation of the individual steps (i.e., software objects) of the steps program and their execution order clearly anticipates a prototype (i.e., sequence of operation). Furthermore, col.3:46-52 of Limondin specifically reads, "In addition to *describing sequences of vision operations*, the disclosed *step program* architecture can also be used to describe a variety of other cell control operations which often have to be performed in conjunction with vision processing. Such programs include but are not limited to I/O, communications, and *motion control applications*" (Emphasis added). Needless to say, Limondin's step program that is generated from a sequence of motion control operations for a motion control application clearly anticipates "generating a graphical program based on the prototype (i.e., sequence of motion control operations)".

Argument 3 (with respect to Claims 1, 25, 26, 28, 37, 39, 41-45): "Limondin does not teach the use of DAQ operations or DAQ measurement devices" (Brief, page 20, 2nd full paragraph).

Response to Argument 3: It has been noted that Appellants assert, "DAQ acronym is not simply an abbreviation for the words 'data acquisition', but refers to 'specific instrumentation technology for acquiring measurement data'" (Emphasis added) (Brief, page 20, 2nd full paragraph). However, contrary to Appellants' assertion, "DAQ" is first and foremost, the abbreviation for the words 'data acquisition'. Thus, "DAQ" generally refers to a plurality of "data acquiring" techniques. In other words, contrary to Appellants' argument, the acronym *DAQ does not refer to any one specific instrumentation technique alone*. As such, the term "DAQ" can only refer to a "data acquiring" aspect of a 'data acquiring' system and/or process. Moreover, throughout the claims, the acronym "DAQ" is used as an adjective (i.e., not able to stand alone on its own), describing the operations/functions. As claimed, DAQ merely refers to the "data acquiring" aspect of said operations/functions. As established above in Response to Argument 2, col.3:46-52 of Limondin specifically reads, "In addition to *describing sequences of vision operations*, the disclosed *step program* architecture can also be used to describe a variety of other cell control operations which often have to be performed in conjunction with vision processing. Such programs include but are not limited to I/O, communications, and *motion control applications*" (Emphasis added).

Needless to say, DAQ (i.e., data acquisition technology) is inherently employed in said vision operations and said motion control applications.

Argument 4 (with respect to claim 5): "Limondin teaches nothing whatsoever about a step program comprising both a block diagram portion and a user interface panel portion" (Brief, page 21, 2nd full paragraph).

Response Argument 4: It should be noted that page 12 (lines 4-7) of the Specification defines "block diagram" (also referred to as "graphical program") to be "a program comprising graphical code, e.g., two or more interconnected nodes or icons, wherein the interconnected nodes or icons may visually indicate the functionality of the program". As acknowledged by Appellants (Brief, page 21, 1st full paragraph), FIG.2 and associated text (see at least col.5:37-54) of Limondin explicitly discloses displaying step program 100, wherein each iconic (i.e., graphical) picture represents an individual step operation (e.g., InspectionStep 10, SnapshotStep 110, AcquireStep 112, etc) (i.e., block diagram containing two or more interconnected nodes/icons) of step program 100. Col.5:21-24 of Limondin specifically reads, "The execution order is encoded by the step program 100 first by the position of the step in the tree, i.e. in the order one would enumerate them by reading their name from left-to-right and top-to-bottom". Thus the displayed step program 100 in FIG.2 of Limondin clearly anticipates the "graphical program including a block diagram portion". Col.9:17-21 and FIG.9 of Limondin specifically disclose displaying the relationship (i.e., interconnection) between the output results generated by a step program". Moreover, col.4:56-60 of Limondin specifically

Art Unit: 2192

reads, "each step and datum in a step program contains a special identifier or GUI (Globally Unique Identifier) which allows the step or datum to dynamically create the editor objects that let a user change parameters and settings and train a step or datum" (Emphasis added). In other words, the editor objects representing the steps (i.e., operations) are dynamically created (i.e., during execution of the step program) to enable real time modification of the steps and/or their execution order. As such, it is inherent that a "user interface panel portion" is included in (i.e., created by) the step (i.e., graphical) program. For without the "user interface panel portion" that processes graphical user inputs (i.e., user graphically editing of the steps and data), it would be impossible to facilitate real time graphical modification of the step program let alone the development and implementation of the step program (e.g., machine vision or motion control application) without consideration for the underlying programming language.

Argument 5 (with respect to Claims 6 and 31): "Limondin's step program is not a graphical data flow program and does not visually indicate data flow among the nodes" (Brief, page 22, 1st full paragraph).

Response to Argument 5: First, it should be noted that "visually indicate data flow among the nodes" is not required by the plain language of the claims. Furthermore, the plain language of the claims do not require "nodes in the graphical program to be interconnected by lines or wires that visually indicate data flow" (Emphasis added) as argued by Appellants (Brief, page 21, last paragraph). Moreover, it is submitted that Limondin's step program clearly anticipates a graphical data flow program because as

Art Unit: 2192

established above, col.5:21-27 of Limondin specifically reads, "The *execution order* (i.e., data flow format) is encoded by the step program 100 first by the position of the step in the tree, i.e., in the order one would enumerate them by reading their name from left-to-right and top-to-bottom. This is referred as walking or traversing the tree making sure each position is visited once until all steps have been traversed". Col.8:55-57 of Limondin further reads "the *_refto* field of every datum object handles connections between inputs of one step and outputs of another". Since, as established above, every step and every datum in the step program can dynamically create editor (graphical) objects that let a user change parameters and settings and train a step or datum (col.4:56-60), Limondin clearly anticipates "a graphical data flow program wherein the nodes are connected according to a data flow format".

Argument 6: For claims 7-10 (Brief, pages 22-24), Appellants repeat the same argument as Argument 1, thus are respectfully referred to Response to Argument 1 established above.

Argument 7 (with respect to claim 11): "Limondin does not teach the limitations recited in claim 11" (Brief, page 25, 3rd paragraph).

Response to Argument 7: As acknowledged by Appellants (Brief, page 25, 2nd paragraph), col.4:55-61 of Limondin specifically reads, "each step and datum in a step program contains a special identifier or GUID which allows the step or datum to dynamically create the editor objects that let a user change parameters and settings

Art Unit: 2192

and train a step or datum. The GUID is stored on the host in a special *database*" (Emphasis added). Col.4:64-col.5:2 of Limondin further reads, "The host provides standard operating system APIs to create an object dynamically given its GUID. This, in essence generates a dual object hierarchy where every runtime step or datum may contain a pointer to an editor object providing a user interface for manipulating the step parameters and inputs". Col.5:8-12 further reads, "On the target, these identifiers (i.e., GUIDs stored in the steps and data) are either ignored when the step program is controlled by the host or are used to generate runtime user interface editors to manipulate the step inputs when the machine vision step program is running in a standalone configuration". It is clear from these passages that each step (i.e., operation) is associated with a GUID that is stored in the database. It is also clear that when the step program is controlled by the host, the editor objects are created dynamically by utilizing their GUIDs (stored and retrieved from the host database). Thus, contrary to Appellants' argument, the GUIDs stored in the database, which are used to create editor objects at runtime clearly anticipate the limitations of claim 11.

Argument 8 (with respect to claims 21 and 36): Appellants merely allege that Limondin does not teach limitations of claims 21 and 36 without adequately analyzing the claim rejections (Brief, page 25, last paragraph). As clearly established in the Final Office Action mailed on 11/28/2005 (pages 13-14), claims 21 and 36 recite the same limitations clearly addressed in rejection of claim 19, for which Appellants did not

Art Unit: 2192

dispute. As such, the Examiner maintains that Limondin anticipates the limitations recited in claims 19, 21 and 36.

Argument 9 (with respect to claim 38): Appellants merely refer to the previous arguments and allege that Limondin does not disclose the limitations of claim 38 (Brief, page 26). Thus, the previous Responses to Arguments apply here.

Argument 10 (with respect to claim 24): Appellants repeat Argument 4 (with respect to claim 5), i.e., Limondin does not teach “displaying a graphical (user interface) panel [portion] for setting properties of a step and receiving user input ...” (Brief, page 28).

Response Argument 10: Col.9:17-21 and FIG.9 of Limondin specifically disclose displaying the relationship (i.e., interconnection) between the output results generated by a step program. Moreover, col.4:56-60 of Limondin specifically reads, “each step and datum in a step program contains a special identifier or GUI (Globally Unique Identifier) which allows the step or datum to dynamically create the editor objects *that let a user change parameters and settings and train a step or datum*” (Emphasis added). In other words, the editor objects representing the steps (i.e., operations) are dynamically created (i.e., during execution of the step program) to enable real time modification of the steps and/or their execution order. As such, it is inherent that a “user interface panel portion” is included in (i.e., created by) the step (i.e., graphical) program.

Art Unit: 2192

For without the "user interface panel portion" that processes graphical user inputs (i.e., user graphically editing of the steps and data), it would be impossible to facilitate real time graphical modification of the step program let alone the development and implementation of the step program (e.g., machine vision or motion control application) without consideration for the underlying programming language.

Argument 11 (with respect to 103 rejection): "McDonald does not teach creating an association between the sequence of operations specified by a user and graphical program automatically generated based on the sequence of operations" (Brief, pages 28-29).

Response to Argument 11: It should be noted that one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). In other words, McDonald was not relied upon to teach "automatically generating the graphical program based on the user specified sequence of operations". Rather, this limitation, as has been established, is already anticipated by Limondin (see Response to Argument 1). Furthermore, Appellants also acknowledge (Brief, page 28), "McDonald teaches creating an association between graphical code (e.g., Limondin's step program) and a control, e.g., a user interface control (e.g., step node containing child step nodes aka "sequence of operations"). As such, McDonald clearly suggests creating an

Art Unit: 2192

association between the Limondin's sequence of operations specified by a user and graphical program automatically generated based on the sequence of operations.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

Art Unit: 2192

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

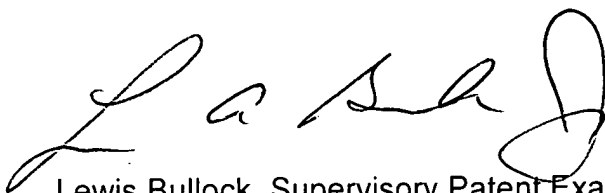
Chrystine Pham /Chrystine Pham/

A handwritten signature in black ink, appearing to read 'Tuan Dam', with a stylized, flowing script.

**TUAN DAM
SUPERVISORY PATENT EXAMINER**

Conferees:

Tuan Dam, Supervisory Patent Examiner, Art Unit 2192

A handwritten signature in black ink, appearing to read 'Lewis Bullock', with a stylized, flowing script.

Lewis Bullock, Supervisory Patent Examiner, Art Unit 2193